Multi-concept document classification using a perceptron-like algorithm

Clay Woolam
University of Texas at Dallas
cpw021000@utdallas.edu

Latifur Khan
University of Texas at Dallas
lkhan@utdallas.edu

Abstract

Previous work in hierarchical categorization focuses on the hierarchical perceptron (Hieron) algorithm. Hierarchical perceptron works on the principals of the perceptron, that is each label in the hierarchy has an associated weight vector. To account for the hierarchy, we begin at the root of the tree and sum all weights to the target label. We make a prediction by considering the label that yields the maximum inner product of its feature set with its path-summed weights. Learning is done by adjusting the weights along the path from the predicted node to the correct node by a specific loss function that adheres to the large margin principal. There are several problems with applying this approach to a multiple class problem. In many cases we could end up punishing weights that gave a correct prediction, because the algorithm can only take a single case at a time. In this paper we present an extended hierarchical perceptron algorithm capable of solving the multiple categorization problem (MultiHieron). We introduce a new aggregate loss function for multiple label learning. Like the basic Hieron algorithm margin requirements are established and it is shown that the loss function is bounded so long as our feature set fulfills said margin requirements. Then, significant improvement over the basic Hieron algorithm is demonstrated on the Aviation Safety Reporting System (ASRS) flight anomaly database and OntoNews corpus using both flat and hierarchical categorization metrics.

1. Introduction

The impetus behind semantic web research remains the vision of supplementing availability with utility; that is, the World Wide Web provides availability of digital media, but the semantic web will allow presently available digital media to be used to serve new purposes, an example of which is semantic level information retrieval.

The semantic web is an extension of today's Web technology; it boasts the ability to make Web resources accessible by their semantic contents rather than merely by keywords and their syntactic forms. Due to its well-established mechanisms for expressing machine-interpretable information, information and Web services previously available for human consumption can be created in a well-defined, structured format from which machines can comprehend, process and interoperate in an open, distributed computing environment.

At present, there are vast amounts of digital media available on the web. Once this media gets associated with machine-understandable metadata, the web can serve as a potentially unlimited supplier for documents, which could populate themselves by searching via keywords and subsequently retrieving articles. One of the greatest challenges, so much that it has become one of the main criticisms, in moving toward the semantic web is the generation of this metadata. In order for metadata creation to be feasible, an automatic (semi-automatic) method is required because manual metadata creation is not scalable over large quantities of information.

An ontology is a specification of an abstract, simplified view of the world that we wish to represent for some purpose [5] and [4]. Therefore, an ontology defines a set of representational terms that we call concepts. Interrelationships among these concepts describe a target world. Information extraction (IE) is a process of extracting structured machine-understandable information from documents (metadata). Ontology based information extraction (OBIE) aims to automatically extract information from text to concepts in an ontology. OBIE is a critical part of building a useful semantic web. Without it, annotating and collecting old data would be an extremely costly process as humans would have to do it. From the given ontology we would like to generate relevant concepts for documents. This can be done by applying the hierarchical perceptron based algorithm.

In this paper, we further investigate a large margin hierarchical perceptron algorithm developed in [2]. Later Yaoyong and Bontcheva demonstrated promising results by applying this algorithm to the OBIE task in [7]. They accomplished this by training multiple classifiers, two for each class in the tree and adding an extra class to the hierarchy to capture non relevant tokens. Previous work with this algorithm focused on the single concept problem, for example a document attached to a single class. If a document is multi-label (attached to multiple classes), the algorithm may not work well.

In our case, we have the Aviation Safety Reporting System (ASRS) flight anomaly database ([9]) and OntoNews corpus ([7]). The ASRS database consists of a set of free-form text documents, each having multiple labels which belong to a structured hierarchy (see fig 2). The OntoNews database has been previously studied and serves as our base test case.

Previous work in hierarchical classification has focused on single label learning. The ASRS database has motivated us to apply hierarchical classification techniques to multi-label learning. The original hierarchical perceptron, Hieron, worked by training a weight vectors corresponding to a labels in a hierarchy. Prediction is done by reporting the path to a label that yields the highest inner product of the features in the sample with the summation over all weights in the path of each label. Training is then done by adjusting relevant labels. We build upon this idea to make an algorithm that can learn documents that have multiple labels attached to them. The main contribution of this paper is to introduce a new multi-label update rule, analytically describe it by bounding the loss function, and empirically show that this approach is good.

2. Hierarchical perceptron algorithm

The hierarchical perceptron algorithm, named Hieron, was first introduced by Dekel et al ([2]). It was presented as two algorithms, an Online Hieron that came with a thorough explanation and detailed analysis and a Batch Hieron, a few modifications on top of the Online Hieron that yielded better empirical results. This paper focuses on improvement in the Online Hieron algorithm. Incidentally, the Batch Hieron algorithm did not preform as well as Online Hieron when applied to the multi-category problem (using the methods described in section 5 of this paper).

We start by arranging our set of labels, Y, in a structured hierarchy and giving each label an associated set of weights $\mathbf{w}^v \in \mathbb{R}^n$ where $v \in Y$. Also, for each label, $\mathbf{W}^v = \sum_{u\in P(v)} \mathbf{w}^u$, where $P(v)$ is a set of nodes along the path from the root to v. Finally, predictions are made using the following rule,

$$f(x) = \arg\max_{v\in Y} \mathbf{W}^v \cdot \mathbf{x} \tag{1}$$

Dekel et al. also describe the online Hieron algorithm, an efficient way to learn these weights. All weights are initialized to zero. During each round, denoted by timestep i, we query the system for a prediction,

$$\hat{y}_i = \arg\max_{v\in Y} \sum_{u\in P(v)} \mathbf{w}_i^v \cdot \mathbf{x}_i \tag{2}$$

The resulting error is called the tree distance, denoted by $\gamma(y_i, \hat{y}_i)$. The algorithm then assumes that there exists a set of weights $\{\omega^v\}_{v\in y}$ such that for all instance-label pairs $(\mathbf{x}_i, y_i)$ in the training set and $r \in Y\backslash\{y_i\}$, the following holds

$$\sum_{v\in P(y_i)} \omega^v \cdot \mathbf{x}_i - \sum_{u\in P(r)} \omega^u \cdot \mathbf{x}_i \geq \sqrt{\gamma(y_i, r)} \tag{3}$$

From this, we have a loss function that forms the basis for our update,

$$L(\{\mathbf{w}^v\}, \mathbf{x}_i, y_i) = \sum_{v\in P(y_i)} \omega^v \cdot \mathbf{x}_i - \sum_{u\in P(r)} \omega^u \cdot \mathbf{x}_i + \sqrt{\gamma(y_i, r)} \tag{4}$$

Algorithm 1 Online Hieron

```
Initialize: $\forall v \in Y : \mathbf{w}_0^v = \mathbf{0}, i = 0$
1. for $i = 1$ to $m$
Using $(\mathbf{x}_i, \mathbf{y}_i)$
2. Predict: $\hat{y} = \arg\max_{y\in Y} \sum_{v\in P(y)} \mathbf{w}_i^v \cdot \mathbf{x}_i$
3. Update:
$w_{i+1}^v = w_i^v + \alpha_i \mathbf{x}_i$, if $v \in P(y)\backslash P(\hat{y})$
$w_{i+1}^v = w_i^v - \alpha_i \mathbf{x}_i$, if $v \in P(\hat{y})\backslash P(y)$
where $\alpha_i = \frac{L(\{\mathbf{w}_i^v\}, \mathbf{x}_i, y_i)}{\gamma(y_i, \hat{y})||\mathbf{x}_i||^2}$
```

Algorithm 1 is the basic Online Hieron algorithm that has been subject to formal analysis. Other similar algorithms based on the same concept exist, such as the aforementioned Batch Hieron, but this paper will focus on using the same underlying concept of the algorithm to solve a fundamentally different problem. In line 1, we iterate through our entire training set, which has m training instances. On each iteration we have a new $(\mathbf{x}_i, \mathbf{y}_i)$. In line 2, a prediction is made. If it's correct, no update is made on line 3, if false, then only relevant nodes are updated. Updates are made using a update rule derived by Dekel et al.[2]

3 Multi-category hierarchical perceptron

As mentioned, the Hieron algorithm has shown promising results for the single label categorization problem. There are many cases, especially with regards to ontologies, where we need to do multi-category prediction. The same prediction model still holds, the prediction simply need be refined to include the highest n resulting path-instance products.

Unfortunately, learning using the ordinary Hieron algorithm had a few problems. Initially we used learning algorithm 1, however, intuition tells us that there are a few problems with this method. First, the problem that the original algorithm was attempting to solve was framed in terms of one prediction label for one true label, not many labels, so our update rule may not work well. Second, we could have a case where we erase important information when updates are made for different labels on the same document as the following example shows.

Imagine we have a document that has 2 labels. Depicted in 1 is such a document following two different weight update rules. Fig 1(a), shows how we may have an undesirable update case by repeating predictions. The bottom right node represents the correct label (y) and its sibling represents the predicted node ($\hat{y}$). This is incorrect, so the weights on the true node will be rewarded (denoted by $+$ in fig 1(a)) while the weights on the prediction will be punished (denoted by $-$ in fig 1(a)). Because they share the same parent, no other nodes need be updated. When queried for another prediction, it makes what would be the correct label in step 1, but is the incorrect label for step 2. Here, the left-most leaf node and second right most leaf node are predicted and the true class, respectively (Tree in fig 1(a)). Updating now cancels out half of the information we added to the system in step 1. It is also expensive, 6 nodes have been updated in the process of training one document.

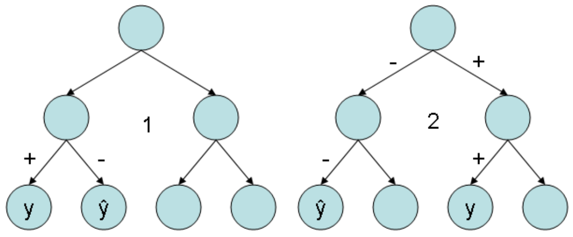
In 1(b), we can get more than one prediction at one time and only 4 updates are made, on each of leaf node in the tree. Both predictions are incorrect, but appear as siblings of the true values, therefore only those nodes need be adjusted and the rest of the tree remains untouched. In other words, left most leaf and second right most leaf nodes are rewarded; second most leaf and right most leaf nodes are punished. As the following sections will demonstrate, not only does this method intuitively feel better, but it is advantageous in several ways. By updating everything in one shot, we are taking into account all information available for a more complete result. Also, by making multiple predictions at once we end up making fewer weight updates.

3.1 Proposed algorithm

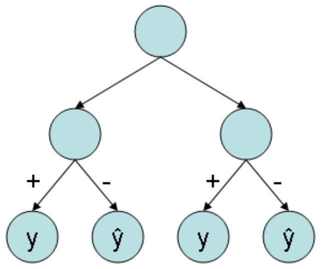
To develop a better algorithm, the problem must be redefined in multi-category terms. The new algorithm will assume that, for all instance-label pairs $(\mathbf{x}_i,\mathbf{y}_i)$ in the training set and $\mathbf{r} \subset Y$ such that at least one $r_j \in Y \backslash \mathbf{y}_i$, the following margin requirements hold

$$\sum_{z\in\mathbf{y}_i}\sum_{v\in P(z)} \omega^v \cdot \mathbf{x}_i - \sum_{q\in\mathbf{r}}\sum_{u\in P(r)} \omega^u \cdot \mathbf{x}_i \geq \sqrt{\gamma(\{\mathbf{y}\},\{\mathbf{r}\})} \qquad (5)$$

From here, and taking a similar approach to the method



(a) Original Hieron learning.



(b) MultiHieron learning.

Figure 1. Example of learning weights for the same document containing two labels.

used in [2], we will craft MultiHieron, the multi-category hierarchical perceptron algorithm. First, notice that our tree distance function has changed characteristics slightly. We now denote $\gamma(\mathbf{V},\mathbf{U})$ to be

$$\gamma(\mathbf{V},\mathbf{U}) = |\bigcup_{v\in\mathbf{V}} P(v) \triangle \bigcup_{u\in\mathbf{U}} P(u)| \qquad (6)$$

Note that $\triangle$ denotes the symmetric difference ($A \triangle B = (A\backslash B)\bigcup(B\backslash A)$) between sets. Following this, our loss function becomes

$$L(\{\mathbf{w_i^v}\},\mathbf{x}_i,\mathbf{y}_i) = \sum_{y\in\mathbf{y}_i}\sum_{v\in P(y)} \mathbf{w}_i^v \cdot \mathbf{x}_i - \sum_{r\in\mathbf{r}}\sum_{u\in P(r)} \mathbf{w}_i^u \cdot \mathbf{x}_i + \sqrt{\gamma(\mathbf{y}_i,\mathbf{r})} \qquad (7)$$

To control learning by ensuring that the margin requirement is met but also keeping the updated weights close to the previous, we follow the method provided in [2] and say,

$$\min_{\{\mathbf{w}^v\}} \frac{1}{2}\sum_{v\in Y} ||\mathbf{w}^v - \mathbf{w}_i^v||^2$$
$$s.t. \sum_{y\in\mathbf{y}_i}\sum_{v\in P(y)} \mathbf{w}^v \cdot \mathbf{x}_i - \sum_{\hat{y}\in\hat{\mathbf{y}}_i}\sum_{u\in P(\hat{y})} \mathbf{w}^u \cdot \mathbf{x}_i \geq \sqrt{\gamma(\mathbf{y}_i,\hat{\mathbf{y}}_i)} \qquad (8)$$

We can solve this condition using Lagrange multipliers, α_i

and preform some optimization to get:

$$\alpha_i = \frac{L(\{\mathbf{w}_i^v\}, \mathbf{x}_i, y_i)}{\gamma(\mathbf{y}_i, \hat{\mathbf{y}}_i)||\mathbf{x}_i||^2} \tag{9}$$

Algorithm 2 Online MultiHieron

```
Initialize: $\forall v \in Y : \mathbf{w}_0^v = \mathbf{0}, i = 0$
1. for $i = 1, 2, ..., m$:
Using: $(\mathbf{x}_i, \mathbf{y}_i)$
2. Predict: $\hat{\mathbf{y}} = \arg\max_{y \in Y}(|\mathbf{y}_i|) \sum_{v \in P(y)} \mathbf{w}_i^v \cdot \mathbf{x}_i$
3. Update:
$w_{i+1}^v = w_i^v + \alpha_i \mathbf{x}_i$, if $v \in \bigcup_{y \in \mathbf{y}_i} P(y)$
$w_{i+1}^v = w_i^v - \alpha_i \mathbf{x}_i$, if $v \in \bigcup_{\hat{y} \in \hat{\mathbf{y}}_i} P(\hat{y})$
$\alpha_i = \frac{L(\{\mathbf{w_i^v}\}, \mathbf{x}_i, y_i)}{\gamma(\mathbf{y}_i, \hat{\mathbf{y}}_i)||\mathbf{x}_i||^2}$
```

Algorithm 2 is similar to algorithm 1. It iterates through every one of the m instances in the training set in line 1. During each iteration, line 2 gets $|\mathbf{y}_i|$ predictions, and 3 preforms weight updates if necessary. The main difference between the two algorithms in line 3, where we are performing updates on multiple labels. Our update rule contains a different path symmetric difference formula, defined as eq. 6. Also, we update each relevant prediction and true class.

4 Analysis of MultiHieron

MultiHieron is not any sort of a specialized multi-category version of the Hieron algorithm, it is actually more general than the original. It can be trivially shown that MultiHieron will reduce to Hieron on any single label categorization problem. If $\mathbf{y}_i$ and $\mathbf{r}$ have maximum size of 1 for all i, then $\gamma(\mathbf{y}_i, \mathbf{r}) = \gamma(y_i, r)$, the symmetric difference of the path to only two labels. All training, updates, and predictions will remain the same.

In [2], Dekel et al. provide a theorem that implied that the cumulative loss suffered by online Hieron is bounded as long as the margin requirements are satisfied. The following is a proof of the theorem created to state this as fact.

Theorem 4.1. *Let $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$ be a sequence of examples where $\mathbf{x}_i \in \mathbb{R}^n$ and $\mathbf{y}_i \in Y$. Assume there exists a set $\{\omega^v : \forall v \in y\}$ that satisfies equation 5 for all $1 \leq i \leq m$. Then, the following holds,*

$$\sum_{i=1}^m L^2(\{\mathbf{w}_i^v\}, \mathbf{x}_i, \mathbf{y}_i) \leq \sum_{v \in Y} ||\omega^v||^2 \gamma_{max} R^2 \tag{10}$$

where for all i, $||\mathbf{x}_i|| \leq R$ and $\gamma(\mathbf{y}_i, \hat{\mathbf{y}}_i) \leq \gamma_{max}$.

Proof. Define $\bar{\omega}$ to be a concatenation of all vectors in $\{\omega^v\}$ and, likewise, $\bar{\mathbf{w}}_i$ in the same manner. The squared distance, δ_i is

$$\delta_i = ||\bar{\mathbf{w}}_i - \bar{\omega}||^2 - ||\bar{\mathbf{w}}_{i+1} - \bar{\omega}||^2$$

Now we can get upper bounds and lower bounds on δ_i over all i by,

$$\begin{aligned}\sum_{i=1}^m \delta_i &= \sum_{i=1}^m ||\bar{\mathbf{w}}_i - \bar{\omega}||^2 - ||\bar{\mathbf{w}}_{i+1} - \bar{\omega}||^2 \\ &= ||\bar{\mathbf{w}}_1 - \bar{\omega}||^2 - ||\bar{\mathbf{w}}_m - \bar{\omega}||^2 \\ &\leq ||\bar{\mathbf{w}}_1 - \bar{\omega}||^2 \\ &\leq ||\bar{\omega}||^2 = \sum_{v \in Y} ||\omega^v||^2\end{aligned}$$

Thus we have our an upper bound on all $\sum_{i \in [1,m]} \delta_i$.

To get our lower bound, use the minimizer of equation 8 producing a result $\bar{\mathbf{w}}_{i+1}$. Using a theorem (Censor & Zenios, 1997, Thm. 2.4.1) [1], we have the following inequality,

$$||\bar{\mathbf{w}}_i - \bar{\omega}||^2 - ||\bar{\mathbf{w}}_{i+1} - \bar{\omega}||^2 \geq ||\bar{\mathbf{w}}_i - \bar{\mathbf{w}}_{i+1}||^2$$

So, $\delta_i \geq ||\bar{\mathbf{w}}_i - \bar{\mathbf{w}}_{i+1}||^2$. After updates, only weights $\mathbf{w}_i^v$ are updated if $v \in \bigcup_{y \in \mathbf{y}_i} P(y) \triangle \bigcup_{\hat{y} \in \hat{\mathbf{y}}_i} P(\hat{y})$, which means,

$$\begin{aligned}||\bar{\mathbf{w}}_i - \bar{\mathbf{w}}_{i+1}||^2 &= \sum_{y \in Y} ||\mathbf{w}_i - \mathbf{w}_{i+1}||^2 \\ &= \sum_{v \in \underset{y \in \mathbf{y}_i}{\cup} P(y) \triangle \underset{\hat{y} \in \hat{\mathbf{y}}_i}{\cup} P(\hat{y})} ||\mathbf{w}_i - \mathbf{w}_{i+1}||^2\end{aligned}$$

Now we can use the update rule to get,

$$\begin{aligned}\sum_{v \in \underset{y \in \mathbf{y}_i}{\cup} P(y) \triangle \underset{\hat{y} \in \hat{\mathbf{y}}_i}{\cup} P(\hat{y})} ||\mathbf{w}_i - \mathbf{w}_{i+1}||^2 &= \sum_v \alpha_i^2 ||\mathbf{x}_i||^2 \\ &= \gamma(\mathbf{y}_i, \hat{\mathbf{y}}_i) \alpha_i^2 ||\mathbf{x}_i||^2\end{aligned}$$

Plugging in equation 9,

$$\begin{aligned}\delta_i &\geq \frac{L^2(\{\mathbf{w}_i^v\}, \mathbf{x}_i, \mathbf{y}_i)}{\gamma(\mathbf{y}_i, \hat{\mathbf{y}}_i)||\mathbf{x}_i||^2} \\ &\geq \frac{L^2(\{\mathbf{w}_i^v\}, \mathbf{x}_i, \mathbf{y}_i)}{\gamma_{max} R^2}\end{aligned}$$

So,

$$\sum_{i=1}^m \frac{L^2(\{\mathbf{w}_i^v\}, \mathbf{x}_i, \mathbf{y}_i)}{\gamma_{max} R^2} \leq \sum_{i=1}^m \delta_i \leq \sum_{v \in Y} ||\omega^v||^2$$

Finally, simply multiplying both sides by γ_{max} and R, we have our original claim. □

5 Experimentation

In experimentation, two datasets were used, the OntoNews corpus and the ASRS flight anomaly database. The

ASRS flight anomaly database will be discussed in great detail in section 5.1. The OntoNews corpus is a collection of 290 ontology annotated news reports (see [8] and [7] for more information about this corpus). These were annotated using the Proton ontology. [10]

Previously, Hieron had shown good results at performing ontology-based information extraction on this dataset. Because it has produced good results before, we are using it as our base set, but in a different way. Instead of preforming information extraction, we treat the documents as they are treated in the ASRS database. That is, we convert each news report to tf-idf weights and treat all annotations as labels for the document. For example, if a document discussed a football game, it would mention a Player, SportsBuilding, SportsTeam, etc., that would all be treated as labels for the document. The dataset was divided into 250 training documents and 40 test documents.

5.1 Description of ASRS anomaly database

The Aviation Safety Reporting System (ASRS) database is a repository of voluntary, confidential safety information provided by aviation personnel of all ranks, including pilots, controllers, mechanics, flight attendants and dispatchers. The database includes almost 150,000 incident reports submitted over more than 30 years. It has two major characteristics that distinguish it from other datasets: a document may have multiple anomalies (multi-categories/multi-labels) and each category belongs in a structured hierarchy. The difficulties associated with categorizing the documents as highly unstructured free form texts was addressed previously in [8], however, they lost a significant amount of highly relevant information by neglecting the underlying hierarchical structure of the categorization set. The ASRS database has 13 major classes, which are fairly general observations such as "inflight encounter" or "conflict", followed by 55 sub classes, which are much more specific. A report might be labeled as both "inflight encounter : weather" and "cabin event : passenger misconduct", if, say, it was a report about a passenger acting particularly angrily about the weather disturbing his flight.

In experimentation, a subset of the documents were chosen at random and tf-idf weights were generated to transform the document to vector forms. Then the feature space was determined through entropy calculations. Figure 3 shows the properties of the database. Our feature space has been restricted to 3814 distinct words, chosen by entropy analysis

Training instances	20000
Testing instances	10000
Features per instance	3814
Minimum labels per instance	1
Maximum labels per instance	10
Average labels per instance	2.71
Maximum depth of hierarchy	3
Number of leaf nodes	55

Figure 3. ASRS selected dataset information

5.2 Experimental setup

Testing utilized a **threshold based** method of analysing prediction accuracy. Both systems were set up in a way such that, when queried with an instance $\mathbf{x}$, it calculates path-weight sums (see eq 2) and return a list of labels, $\mathbf{l}$, and path sums, $\mathbf{p}$ sorted from most likely (highest weight) to least likely (lowest weight). Both sets contain two subsets $\mathbf{l}_t(\mathbf{p}_t)$ and $\mathbf{l}_f(\mathbf{p}_f)$, corresponding to the sets of values predicted as true and ones predicted as false, respectively. These path sums were then normalized to values in $[0, 1]$, such that $\sum_{p \in \mathbf{p}} p = 1$. Next, a threshold value T is defined to have this property,

$$\sum_{p \in \mathbf{p}_t} p \leq T \tag{11}$$

For further analysis, we will use a hierarchical error metric created by Maynard et al. ([8]) called BDM. It is defined as follows: given key node K and response node R, BDM is computed as

$$BDM(K,R) = \frac{BR * CP/n0}{BR * CP/n0 + DPK/n2 + DPR/n3} \tag{12}$$

where **BR** (branching factor) is the average number of branches between the Most Specific Common Abstraction (MSCA) and the key node and response node normalized by average branching factor for the entire hierarchy, **CP** is the shortest path length from root node to MSCA, **DPK(R)** is the shortest path from MSCA to K(R), **n0** is the average chain length of the whole hierarchy, **n2** is the average length of all chains containing K from root, and **n3** is the average length of all chains containing R from root. New augmented precision and recall are defined to be (FP and FN represent false positive and false negative counts)

$$AP = \frac{BDM}{BDM + FP} \tag{13}$$

$$AR = \frac{BDM}{BDM + FN} \tag{14}$$

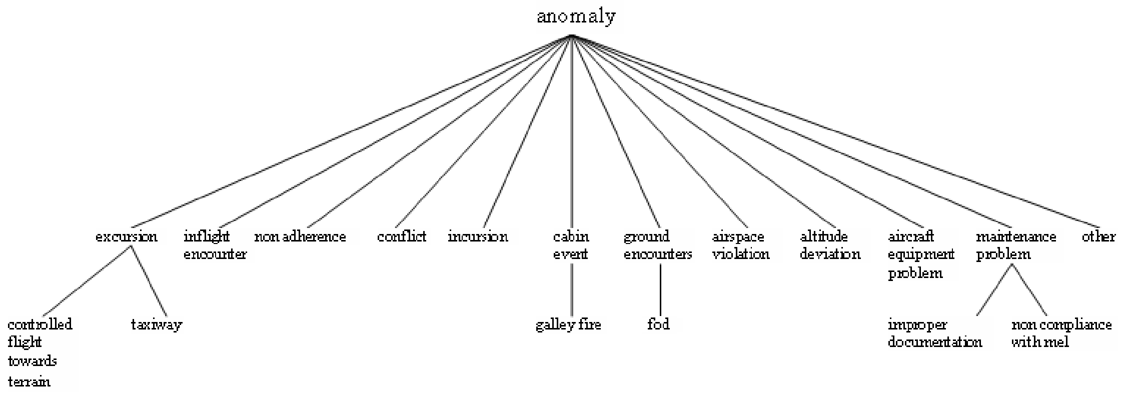


Figure 2. Subset of ASRS database ontology. There are 55 third level nodes, 3 are shown.

Augmented precision and recall gives a better overall picture of an algorithms performance in the hierarchical case because we are accounting for nearness to the true value. In flat classification, we can assign a binary value to each prediction. That is to say, it is either correct, assign a value of 1, or it is not, assign a value of 0. In the hierarchical case, metrics like these allow us to better see what is going on inside the system as it learns. BDM will still give a value of 1 if correct, but also return a real number < 1 that takes into account the dynamics of the tree and the location of correct and predicted values on an error. For example, BDM calculation for graph 1 in figure 1(a) is as follows. Our $BR = 1.43$, $CP = 2$, $DPK = 1$, $DPR = 1$, $n0 = 3$, $n2 = 1$, and $n3 = 1$. This gives a BDM of 0.32, according to eq. 12. In graph 2, we have a different $CP = 1$, $DPK = 2$, and $DPR = 2$, therefore BDM is 0.11.

5.3 Results

By varying the threshold that distinguishes positive predictions from negative predictions, we can calculate a precision/recall curve, seen in figure 4(a). Lets examine the graph by looking at the points where recall is 0.6. Notice how the precision 0.35 for MultiHieron and 0.15 for Hieron. We can conclude that the MultiHieron does a significantly better job at classifying documents in this database that demonstrates superiority of our approach. Compare with figure 4(b), our base set, where a similar behaviour is exhibited.

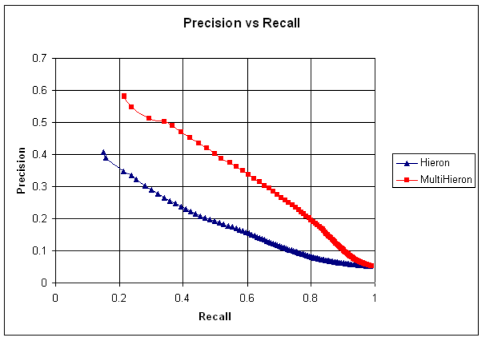
Using the BDM method described above, we generate an augmented precision versus recall curve. Figure 5 gives a dramatic lessening of the distance between the two curves. This is in part due to the flatness of the tree, but it is also apparent that the original Hieron method was good at making predictions that are siblings of the correct label. At an augmented recall of 0.6 in the curve, augmented precision is 0.3 and 0.35 for Hieron and MultiHieron, respectively. MultiHieron is more suited to the task of multiple label learning.

MultiHieron also showed a speedup over Hieron. Figure 6 shows the difference in training time for Hieron and MultiHieron. This speedup almost corresponds to the average 2.7 labels per document given in table 3. Training with multiple labels on the original Hieron algorithm was a multiple step process, with the majority of time spent calculating all of the possible prediction paths. With MultiHieron, only one prediction calculation need be made for each document, therefore the speedup is proportional to the average number of labels per document. Testing uses the exact same algorithms and data structures because the underlying prediction model for the algorithm was not changed, therefore testing speed is the same.

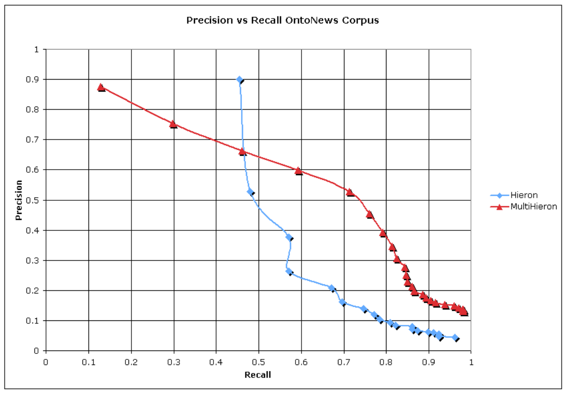
6 Conclusions and future work

We have presented a more generalized hierarchical perceptron algorithm, capable of doing multi-label document training. This new algorithm preserves all of the properties of original Hieron algorithm, and uses the same principles to preform multi-label learning.

The ASRS flight anomaly database posed two unique problems. First, the anomaly reports could have more than one label. Secondly, labels belong to a structured hierarchy. We addressed these problems with the MultiHieron algorithm. MultiHieron showed improvement in both performance and accuracy over Hieron in multi-label learning. The hierarchy was not ideal for our problem. Not only is it very flat and wide, but label paths span the entire hierarchy. Hieron was designed to predict labels for any point in the hierarchy. It is still an important contribution to demonstrate that such a significant improvement can be made with some modifications to the algorithm. Comparison with the OntoNews corpus verified these results.



(a) ASRS database



(b) OntoNews corpus

Figure 4. Precision versus recall for Hieron and MultiHieron

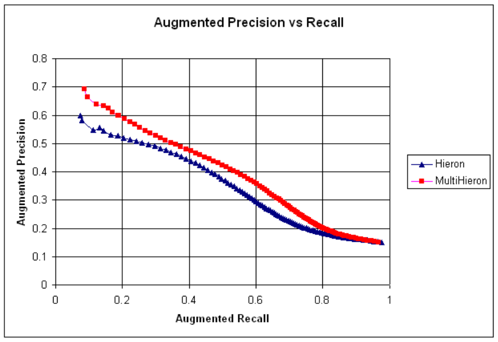


Figure 5. Augmented precision versus recall for Hieron and MultiHieron on ASRS database.

	Hieron	MultiHieron
ASRS Training	605.8s	253.7s
ASRS Testing	79s	79s
OntoNews Training	417.38s	58.69s
OntoNews Testing	7.1s	7.1s

Figure 6. MultiHieron shows a significant performance boost over Hieron for multi-label training

References

[1] Y. A. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms and Applications*. Oxford University Press, March 1997.

[2] O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *ICML '04: Proceedings of the twenty-first International Conference on Machine Learning*, page 27, New York, NY, USA, 2004. ACM.

[3] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. A. Tomlin, and J. Y. Zien. Semtag and seeker: bootstrapping the semantic web via automated semantic annotation. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 178–186, New York, NY, USA, 2003. ACM.

[4] T. R. Gruber. A translation approach to portable ontology specifications. *Knowl. Acquis.*, 5(2):199–220, 1993.

[5] N. Guarino, C. Masolo, and G. Vetere. Ontoseek: Content-based access to the web. *IEEE Intelligent Systems*, 14(3):70–80, 1999.

[6] E. Hyvnen, S. Saarela, A. Styrman, and K. Viljanen. Ontology-based image retrieval. *WWW (Posters)*, 2003.

[7] Y. Li and K. Bontcheva. Hierarchical, perceptron-like learning for ontology-based information extraction. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 777–786, New York, NY, USA, 2007. ACM.

[8] D. Maynard, W. Peters, and Y. Li. Metrics for evaluation of ontology-based information extraction. *WWW 2006 Workshop on "Evaluation of Ontologies for the Web" (EON)*, 2006.

[9] NASA. Aviation safety reporting system database. http://asrs.arc.nasa.gov/search/database.html.

[10] S. Project. Proto ontology (proton). http://proton.semanticweb.org.

[11] A. N. Srivastava, R. Akella, and et. al. Enabling the discovery of recurring anomalies in aerospace problem reports using high-dimensional clustering techniques. In *IEEE Aerospace Conference*, 2006.

[12] A. S. Weigend, E. D. Wiener, and J. O. Pedersen. Exploiting hierarchy in text categorization. *Inf. Retr.*, 1(3):193–216, 1999.

[13] D. Yang and S. A. Zenios. A scalable parallel interior point algorithm for stochastic linear programming and robust optimization. *Computational Optimization and Applications*, 7, January 1997.